A Dynamic Preconditioner for Newton-Krylov Algorithms: Application to Fluid-Structure Interaction

Simone Deparis, Jean-Frédéric Gerbeau, Xavier Vasseur

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

A Dynamic Preconditioner for Newton-Krylov Algorithms. Application to Fluid-Structure Interaction.

Simone Deparis — Jean-Frédéric Gerbeau — Xavier Vasseur

N° 5352

Octobre 2004

Thème BIO

ISSN 0249-6399 ISRN INRIA/RR--5352--FR+ENG

INRIA
ROCQUENCOURT

A Dynamic Preconditioner for Newton-Krylov Algorithms. Application to Fluid-Structure Interaction.

Simone Deparis* , Jean-Frédéric Gerbeau† , Xavier Vasseur‡

Thème BIO — Systèmes biologiques
Projet REO

Rapport de recherche n° 5352 — Octobre 2004 — 25 pages

Abstract: We consider linear and nonlinear convergence acceleration techniques in the framework of Newton or inexact Newton methods. The proposed procedure is based on a new dynamic preconditioner to be used in combination with the GMRES method [20] for reducing the cost of solving a sequence of linear systems. A nonlinear convergence acceleration technique based on a previous work by Washio et al. [25] is also added. The benefits of this combination of acceleration techniques is shown on fluid-structure interaction problems in haemodynamics for two- and three-dimensional applications.

Key-words: preconditionned GMRES, nonlinear acceleration, Krylov methods

* CMCS-IACS EPF Lausanne, CH-1015 Lausanne, Switzerland, deparis@mit.edu
† INRIA Rocquencourt - Projet REO, Jean-Frederic.Gerbeau@inria.fr
‡ Seminar for Applied Mathematics ETH Zürich, CH- 8092 Zürich, Switzerland,vasseur@math.ethz.ch

Unité de recherche INRIA Rocquencourt
Domaine de Voluceau, Rocquencourt, BP 105, 78153 Le Chesnay Cedex (France)
Téléphone : +33 1 39 63 55 11 —Télécopie : +33 1 39 63 53 30

Un préconditionneur dynamique pour des algorithmes de Newton-Krylov. Application à l'interaction fluide-structure.

Résumé : On considère des techniques d'accélération linéaires et non linéaires dans le cadre d'algorithmes de Newton et de Newton inexacts. La méthode d'accélération linéaire proposée, basée sur un nouveau préconditionneur dynamique à utiliser avec GMRES [20], a pour but de diminuer le coût de la résolution d'une série de systèmes linéaires. Elle est complétée par une méthode d'accélération non linéaire basée sur des travaux de Washio *et al.*[25]. L'efficacité de la conjonction des deux techniques est illustrée dans des problèmes d'interaction fluide-structure 2D et 3D provenant de l'hémodynamique.

Mots-clés : GMRES préconditionné, accélération non linéaire, méthodes de Krylov

1 Introduction

Newton–Krylov [1] or Jacobian-free Newton–Krylov methods [12] are popular solution strategies for nonlinear problems in applied mathematics and computational physics. They rely on a combination of Newton-type methods with super-linearly convergence rates and Krylov subspace methods for solving the Newton correction equations. Their main core requires to solve a sequence of linear systems of type:

$$A_i \boldsymbol{x}_i = \boldsymbol{b}_i \text{ (i=1,2, ...)} \tag{1}$$

where the matrices A_i and the right-hand sides $\boldsymbol{b}_i$ change from one system to the next and are not available simultaneously. In this work, we investigate some linear and nonlinear acceleration techniques in the framework of Newton–Krylov or inexact Newton–Krylov algorithms to derive an efficient and robust nonlinear solver.

Our first purpose is to accelerate the convergence of a given linear system by reusing information built during previous resolution processes. Most papers in the literature consider the case of sequence of linear systems with only invariant matrices (e.g. [2, 16, 19, 21, 22]). Moreover if all the right-hand sides are available simultaneously, block methods (e.g. block GMRES [24]) are here relevant. However these techniques do not generalize to the case (1). To our knowledge few attempts have addressed the general case: in [3, 17] relevant techniques restricted to a sequence of systems with symmetric positive definite matrices are proposed, whereas in [15] the nonsymmetric case is considered. The main idea of this latter work is to recycle a subspace that minimizes the loss of orthogonality with a Krylov subspace built in the previous system. The authors propose two approaches based either on augmentation or orthogonalization to reduce the computational cost by recycling. Our technique is possibly one alternative and relies rather on preconditioning. This technique also addresses the general case (1): each matrices A_i are only assumed to be regular and each linear system will be solved with the GMRES method [20]. The proposed dynamic preconditioner consists in exploiting information that is related to the sequence of Hessenberg matrices built during the successive orthogonalization procedures. We show that this new preconditioner is nonsingular and well defined and we describe how to nest more than one preconditioner.

Some aspects of our approach are similar to [4, §4] and [5]. These authors consider a restarted GMRES(m) method and build a preconditioner for the next perturbed system using the Krylov subspace and the Hessenberg matrix stored during the previous GMRES(m) application. They also apply the preconditioners in a nested way to solve the Navier–Stokes equations within a Newton algorithm. Although connected, the two approaches differ in the iterative method used (restarted against full GMRES) and in the stored data: even if the preconditioners are used in a nested way, only one Krylov subspace is stored, hence some informations may be lost. Here we propose to store several Krylov subspaces and the related basis associated to benefit from all the information already collected. The drawback of our approach is the amount of memory needed, however nowadays this item is loosing importance. Moreover, in the fluid-structure interaction problem presented here, the degrees of freedom concerned by the Newton–Krylov algorithm are only those located on the interface; this leads to small sized problems in practice.

We then detail a nonlinear acceleration technique for the resulting preconditioned Newton–Krylov algorithm. Previous works have been done in this respect. For instance Washio et al. [25] and Fokkema et al. [8] have proposed to store both the iterates and the residuals and to search for a better iterate in the affine subspace generated during the previous iterations. Washio et al. [25] have proposed to find the new iterate by minimizing the norm of the linearized residual. Following this approach, we remark that if the linearization of the residual is accurate enough, a new (inexact) Newton step can be carried out even without evaluating the new residual. Indeed, we are able to define an entire acceleration step, where neither the residual nor the inverse of the Jacobian are *explicitly* computed. They are replaced by a linearization of the residual and an application of the dynamic preconditioner respectively.

The resulting algorithm is well suited to problems where the functional or its Jacobian are very expensive to compute. In particular, to build the preconditioner there is no need to explicitly build the matrices A_i (in other words, the Jacobians), but only to evaluate the matrices against given vectors in the GMRES algorithm.

The resulting solution method is then tested in the framework of fluid-structure interaction problems in haemodynamics. The numerical results for two and three-dimensional problems are very satisfactory, with a total computational gain of up to 50% in CPU time versus a standard Newton method.

In section 2 we recall some basic properties of the GMRES algorithm, which forms the basis for the preconditioning approach. Then we define a dynamic preconditioner and prove its well posedness. In section 3 we recall the basic solution method (Newton-Krylov method) and explain the nonlinear convergence acceleration procedure. The resulting solution method is then presented. Finally in section 4 we settle the fluid-structure interaction problem in haemodynamics and we report some numerical results.

2 Dynamic GMRES-based preconditioner

2.1 The GMRES iterative method

Consider the following linear problem: Find $\boldsymbol{x}$ in V such that

$$A\boldsymbol{x} = \boldsymbol{b}, \tag{2}$$

where A is a linear operator from and onto a generic vector space V with scalar product $(\cdot,\cdot)$ and $\boldsymbol{b}$ is an element of V. The GMRES iterative method builds an orthonormal sequence $(\boldsymbol{v}_j)_{j=1,\ldots,k+1}$ in V, and two matrices, one Q_k, orthogonal of size $k+1\times k+1$, and the other R_k, upper triangular of size $k+1\times k$, such that if $\boldsymbol{x} = \sum_{j=1}^{k}\xi_j\boldsymbol{v}_j$, then $A\boldsymbol{x} = \sum_{j=1}^{k+1}\zeta_j\boldsymbol{v}_j$ where $\boldsymbol{\zeta} = Q_kR_k\boldsymbol{\xi}$.

The sequence $(\boldsymbol{v}_j)_{j=1,\ldots,k+1}$ is an orthonormal basis of the Krylov subspace

$$\mathcal{L}_{k+1} = K^{k+1}(A,\boldsymbol{r}_0) = \operatorname{span}\{\boldsymbol{r}_0, A\boldsymbol{r}_0,\ldots,A^k\boldsymbol{r}_0\},$$

where $\boldsymbol{r}_0 = \boldsymbol{b} - A\boldsymbol{x}_0$. Moreover Q_k and R_k are the QR-decomposition of the Hessenberg matrix H, defined by $H_{jl} = (\boldsymbol{v}_j, A\boldsymbol{v}_l)$ (we refer for example to Saad and Schultz [20] and Van der Vorst [23, §4 and §6] for a complete description).

The residual of (2) is minimized over the linear subspace $\mathcal{L}_k$ and the approximated solution is given by

$$\boldsymbol{x} = \arg \min_{\boldsymbol{x} \in \mathcal{L}_k} \|A\boldsymbol{x} - \boldsymbol{b}\|. \tag{3}$$

The problem (3) is equivalent to

$$R_k \boldsymbol{\xi} = \Pi^* Q_k^T \boldsymbol{\beta}, \tag{4}$$

where $\beta_j = (\boldsymbol{b}, \boldsymbol{v}_j)$ for $j = 1, \ldots, k+1$, and Π^* is the projection that sets the last component of a vector to zero (note that the last line of R_k has all components equal to zero). Note that the error is equal to absolute value of the last component of $Q_k^T \boldsymbol{\beta}$.

2.2 Dynamic initial guess

Assume that we have two problems of type (2) on the same space V: Find $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ in V such that

$$A_1 \boldsymbol{x}_1 = \boldsymbol{b}_1 \text{ and } A_2 \boldsymbol{x}_2 = \boldsymbol{b}_2, \tag{5}$$

where the operators A_1, A_2 are close enough, e.g., the first one is a good preconditioner of the second one, but both are expensive to invert. We assume that the first linear system $A_1 \boldsymbol{x}_1 = \boldsymbol{b}_1$ has been solved with GMRES up to a given tolerance. Let denote $k+1$ the number of iterations required. The Krylov subspace $(\boldsymbol{v}_j^{(1)})_{j=1,\ldots,k+1}$ and the QR decompositon of the Hessenberg matrix $Q_{1,k}$ and $R_{1,k}$ are also supposed to be available.

A classical way to define the initial guess for the solution of the second equation $A_2 \boldsymbol{x}_2 = \boldsymbol{b}_2$, is to choose the solution $\boldsymbol{x}_1$ of the first one which is in fact $\arg\min_{\boldsymbol{x} \in \mathcal{L}_{1,k}} \|A_1 \boldsymbol{x} - \boldsymbol{b}_1\|$.

However it is also possible to generate an initial guess for the second problem using the iterates built during the solution of the first linear system. The initial guess can be deduced as follows:

$$\arg \min_{\boldsymbol{x} \in \mathcal{L}_{1,k}} \|A_1 \boldsymbol{x} - \boldsymbol{b}_2\|. \tag{6}$$

Intuitively, this choice seems more appropriate as a first approximation of the solution of $A_2 \boldsymbol{x}_2 = \boldsymbol{b}_2$, since A_1 is supposed to be close to A_2. The minimization in (6) can be carried out without any matrix-vector product involving A_1, since it is equivalent to solve as in (4) the problem reduced to the first k components,

$$R_{1,k}\, \boldsymbol{\xi} = \Pi^* Q_{1,k}^T \boldsymbol{\beta}_2, \tag{7}$$

where the components of $\boldsymbol{\beta}_2$ are now given by $(\boldsymbol{b}_2, \boldsymbol{v}_j^{(1)})$.

2.3 Dynamic preconditioner

We would like to extend this idea by using the information stored in the Krylov subspace and the Q and R matrices obtained during the solution of the previous system to build a preconditioner. The effectiveness of the procedure is based on the assumption that A_1 can be a good preconditioner for A_2.

As suggested in section 2.2, the basic step is to take the orthogonal projection of $\boldsymbol{b}_2$ on the image of $A_1|_{\mathcal{L}_{1,k}}$ (noted $\Pi_{\mathrm{Im}(A_1|_{\mathcal{L}_{1,k}})}\boldsymbol{b}_2$) and then to apply the inverse of $Q_{1,k}R_{1,k}$. Indeed $Q_{1,k}$ and $R_{1,k}$ are the QR factorization of the restriction of A_1 on $\mathcal{L}_{1,k}$

$$A_1|_{\mathcal{L}_{1,k}} : \mathcal{L}_{1,k} \to \mathrm{Im}(A_1|_{\mathcal{L}_{1,k}}) \subset \mathcal{L}_{1,k+1}.$$

Then, to avoid a singular preconditioner (see proof of theorem 2.1), we add $\lambda^{-1}(\boldsymbol{b}_2 - \Pi_{\mathrm{Im}(A_1|_{\mathcal{L}_{1,k}})}\boldsymbol{b}_2)$, where λ is a nonzero scalar to be chosen. The corresponding preconditioner is thus:

$$P_1^{-1} = \left(A_1|_{\mathcal{L}_{1,k}}\right)^{-1} \Pi_{\mathrm{Im}(A_1|_{\mathcal{L}_{1,k}})} + \frac{1}{\lambda}\left(Id - \Pi_{\mathrm{Im}(A_1|_{\mathcal{L}_{1,k}})}\right). \tag{8}$$

Its application to a vector $\boldsymbol{y}$ in V can be carried out in five steps:

$$\begin{aligned}
\zeta_j &= (\boldsymbol{y}, \boldsymbol{v}_j^{(1)}) \quad j = 1, \ldots, k+1, && (9)\\
\boldsymbol{\chi} &= {Q_{1,k}}^T \boldsymbol{\zeta}, && (10)\\
\boldsymbol{\tau} &= Q_{1,k}\,(Id - \Pi^*)\boldsymbol{\chi}, && (11)\\
\boldsymbol{\xi} &: \text{ solve } R_{1,k}\boldsymbol{\xi} = \Pi^*\,\boldsymbol{\chi}, && (12)\\
\boldsymbol{z} &= \sum_{j=1}^{k} \xi_j \boldsymbol{v}_j^{(1)} + \frac{1}{\lambda}\left(\boldsymbol{y} - \sum_{j=1}^{k+1}(\zeta_j - \tau_j)\boldsymbol{v}_j^{(1)}\right), && (13)
\end{aligned}$$

where $\boldsymbol{\zeta} = (\zeta_1, ..., \zeta_{k+1})^T$, etc., and Π^* is the projection that sets the last component of a vector to zero and is equivalent to the projection $\Pi_{\mathrm{Im}(A_1|_{\mathcal{L}_{1,k}})}$ on the image set of $A_1|_{\mathcal{L}_{1,k}}$ (see lemma 2.1). Since the last line of $R_{1,k}$ is zero, equation (12) is well defined and its resolution is straight-forward by backward substitution.

We propose to set the scalar λ equal to the last diagonal element of $R_{1,k}$, i.e., $R_{1,k}(k,k)$. Other choices can be derived from an Aitken relaxation parameter from the previous iterates, see section 4, or from the mean value of the diagonal elements of $R_{1,k}$.

2.4 Invertibility of the preconditioner

To simplify the notation, in this section we refer to A, Q, R, etc., without indices whenever there are no ambiguities.

In the following we prove that for a non singular operator A on a finite dimensional vector field V with scalar product $(\cdot,\cdot)$, the preconditioner defined in (8) has full rank and can be computed following steps (9) to (13).

Theorem 2.1 *If A is not singular, then the operator*

$$M = (A|_{\mathcal{L}_k})^{-1}\,\Pi_{\mathrm{Im}(A|_{\mathcal{L}_k})} + \frac{1}{\lambda}\left(Id - \Pi_{\mathrm{Im}(A|_{\mathcal{L}_k})}\right) \tag{14}$$

is invertible on V.

Proof The GMRES algorithm computes an orthonormal sequence $(\boldsymbol{v}_j)_{j=1,\ldots,k+1}$ and $\mathrm{Im}(A|_{\mathcal{L}_k})$ is included in $\mathcal{L}_{k+1}$. Recall that $\Pi_{\mathrm{Im}(A|_{\mathcal{L}_k})}$ is the orthogonal projection on $\mathrm{Im}(A|_{\mathcal{L}_k})$. Since A is regular, $A|_{\mathcal{L}_k}$ is invertible on $\mathrm{Im}(A|_{\mathcal{L}_k})$ and M is well defined.

Let $\boldsymbol{y}$ be an element in V such that $M\boldsymbol{y} = 0$. We need to show that y is equal to zero. Let $\boldsymbol{y}_2 = \Pi_{\mathrm{Im}(A|_{\mathcal{L}_k})}\boldsymbol{y}$ and $\boldsymbol{y}_1 = \boldsymbol{y} - \boldsymbol{y}_2$. Then

$$\frac{1}{\lambda}\boldsymbol{y}_1 = -\,(A|_{\mathcal{L}_k})^{-1}\,\boldsymbol{y}_2. \tag{15}$$

This implies that $\boldsymbol{y}_1$ is also in $\mathcal{L}_k$, i.e., $\boldsymbol{y}_1 = \sum_{j=1}^{k}\varphi_j\boldsymbol{v}_j$. But since $\boldsymbol{y}_1$ is in the orthogonal hull of $\mathrm{Im}(A|_{\mathcal{L}_k})$, $(\boldsymbol{y}_1, A\boldsymbol{v}_l) = 0$ for all $l = 1, ..., k$, hence

$$0 = \sum_{j=1}^{k}\varphi_j(\boldsymbol{v}_j, A\boldsymbol{v}_l) = \sum_{j=1}^{k}\varphi_j H_{jl},$$

where H is the Hessenberg matrix of A with respect to $(\boldsymbol{v}_j)_{j=1,\ldots,k}$. This can be written in vector form as $\sum_{j=1}^{k}\varphi_j\boldsymbol{h}_j = 0$, where $\boldsymbol{h}_j$ is equal to $(H_{j1}, ..., H_{jk})^T$. Since A is regular and the vectors $\boldsymbol{v}_j$, $j = 1, ..., k$, are linearly independent, the vectors $\boldsymbol{h}_j$ are also linearly independent.

Hence $\varphi_j = 0$ for $j = 1, ..., k$ and $\boldsymbol{y}_1 = 0$. From equation (15) we deduce $\boldsymbol{y}_2 = 0$, which means that $\boldsymbol{y} = 0$ and that M is invertible.

■

Lemma 2.1 *The operator $Q\,\Pi^*\,Q^T$: $\mathbb{R}^{k+1} \to \mathbb{R}^{k+1}$ is an orthogonal projector with respect to the Euclidean scalar product. Moreover, its image is equal to the image of QR: $\mathbb{R}^k \to \mathbb{R}^{k+1}$.*

Proof The operator $Q\,\Pi^*\,Q^T$ is a projection, since

$$\left(Q\Pi^*Q^T\right)^2 = Q\Pi^*\Pi^*Q^T = Q\Pi^*Q^T.$$

To prove the orthogonality first note that $Id - Q\Pi^*Q^T = QQ^T - Q\Pi^*Q^T = Q(Id - \Pi^*)Q^T$. Then for $\boldsymbol{\xi}$ and $\boldsymbol{\zeta}$ in $\mathbb{R}^{k+1}$

$$\left(Q\,(Id - \Pi^*)\,Q^T\boldsymbol{\zeta}\,,\,Q\,\Pi^*\,Q^T\boldsymbol{\xi}\right) = \left((Id - \Pi^*)\,Q^T\boldsymbol{\zeta}\,,\,\Pi^*\,Q^T\boldsymbol{\xi}\right) = 0,$$

since Q is orthogonal. Hence $Q\,\Pi^*\,Q^T$ is an orthogonal projection with respect to the Euclidean scalar product.

We now prove that $\mathrm{Im}\, QR = \mathrm{Im}\, Q\,\Pi^*\,Q^T$. first, $\mathrm{Im}\, QR \subset \mathrm{Im}\, Q\,\Pi^*\,Q^T$, since the last line of R has only zeros and

$$Q\Pi^* Q^T QR = Q\Pi^* R = QR.$$

Then, let $\boldsymbol{\xi} \in \mathbb{R}^k$ and $\boldsymbol{\zeta} \in \mathbb{R}^{k+1}$. The relation $QR\boldsymbol{\xi} = Q\Pi^* Q^T \boldsymbol{\zeta}$ is equivalent to $R\boldsymbol{\xi} = \Pi^* Q^T \boldsymbol{\zeta}$, which has a unique solution since the last line of R has only zeros (R is upper diagonal and all elements of the diagonal are different from zero (else A would be singular)). Hence $\mathrm{Im}\, Q\,\Pi^*\,Q^T \subset \mathrm{Im}\, QR$.

■

Proposition 2.1 *Let $\boldsymbol{y}$ be in V and set $\zeta_j = (\boldsymbol{y}, \boldsymbol{v}_j^{(1)})$, $j = 1, \ldots, k+1$ and let $\boldsymbol{\xi} = Q\,\Pi^*\,Q^T\boldsymbol{\zeta}$. Then*

$$\Pi_{\mathrm{Im}(A|_{\mathcal{L}_k})}\boldsymbol{y} = \sum_{j=1}^{k+1} \xi_j \boldsymbol{v}_j. \tag{16}$$

Proof first note that for $\boldsymbol{x} = \sum_{j=1}^{k} \chi_j \boldsymbol{v}_j$, $A\boldsymbol{x} = \sum_{j=1}^{k+1} (QR\boldsymbol{\chi})_j \boldsymbol{v}_j$. Then, the proof follows from lemma 2.1.

■

2.5 Application to a sequence of problems

In this section we apply the preconditioner defined in the previous section in a nested way when solving a sequence of problems.

The preconditioner may be applied as a left or a right preconditioner. Anyway, we present it in its right form, since a left preconditioned GMRES changes the norm used in computing the residual, while it is not the case in the right preconditioned version. Moreover, it is possible to apply this method to an already existing left preconditioned GMRES, such that the resulting GMRES has both left and right preconditioners (see also [23, §10]).

Suppose that we have solved $A_1\boldsymbol{x} = \boldsymbol{b}_1$ with GMRES and that we define P_1 according to formula (14). Then the second right preconditioned problem reads

$$A_2 P_1^{-1}\,\boldsymbol{y} = \boldsymbol{b}_2 \qquad \boldsymbol{x} = P_1^{-1}\boldsymbol{y}. \tag{17}$$

Suppose now that there is a third linear system to solve. We can use P_1 as a preconditioner, but if operator A_3 is closer to A_2 than to A_1, as in the case of the Newton algorithm, it is more interesting to use P_2. Unfortunately, since we suppose that A_2 (not $A_2P_1^{-1}$!) is a good preconditioner for A_3, this is not possible.

A first cure is to use a technique taken from Flexible GMRES (FGMRES, see [18] or [23]). FGMRES, in addition to the matrices and the basis of the Krylov subspace, stores the basis of the intermediate preconditioned vectors. A work in this direction is under consideration.

Instead, we propose another strategy aiming at nestin the preconditioners. For example, the third preconditioned problem would read:

$$A_3 P_1^{-1} P_2^{-1}\,\boldsymbol{y} = \boldsymbol{b}_3 \qquad \boldsymbol{x} = P_1^{-1}P_2^{-1}\boldsymbol{y}. \tag{18}$$

In fact, P_2 is a preconditioner derived from (17) and is therefore well suited for $A_3 P_1^{-1}$. It is mandatory to keep the number of preconditioners small, so it is recommended to do a restart of the preconditioners. In principle one can either delete all the preconditioners or just keep P_1.

As mentioned above, it is also possible to use an already defined left preconditioner M, such that the problems are

$$M^{-1}A_1\,\boldsymbol{x} = M^{-1}\boldsymbol{b}_1, \tag{19}$$

$$M^{-1}A_2P_1^{-1}\,\boldsymbol{y} = M^{-1}\boldsymbol{b}_2 \qquad \boldsymbol{x} = P_1^{-1}\boldsymbol{y}, \tag{20}$$

$$M^{-1}A_3P_1^{-1}P_2^{-1}\,\boldsymbol{y} = M^{-1}\boldsymbol{b}_3 \qquad \boldsymbol{x} = P_1^{-1}P_2^{-1}\boldsymbol{y}, \tag{21}$$

$$\text{etc.} \tag{22}$$

Indeed, it is possible that the left preconditioner depends on the problem index, as long as the original preconditioned problems, i.e., $M_i^{-1}A_i$, are close to each other.

3 Nonlinear acceleration of the Newton–Krylov algorithm

We present in this section our global procedure for solving nonlinear problems. We first recall the solution method based on Newton–Krylov methods [1]. Then we describe an accelerating procedure that aims at improving its computational efficiency. Finally we present the global framework in an algorithmic fashion.

The main computational challenge is the inversion of the Jacobian matrix (step 3 on this page) with Krylov-type methods. One drawback of this procedure is that in every linearization step the Jacobian must be evaluated in many directions. Nevertheless this potentially huge cost can be reduced by replacing the Jacobian matrix by a simpler and cheaper operator, as done in the inexact Newton algorithm.

In this work a preconditioned GMRES method is used to solve the Jacobian problem stated in step 3. As a preconditioner for GMRES, the dynamic preconditioner presented in section 2 is investigated. An accelerating procedure is presented in the following for enhancing the efficiency of this preconditioned Newton-GMRES procedure.

3.1 Solution method

Given a vector field on $\mathbb{R}^N$, we want to find an approximation $\boldsymbol{x}^*$ in $\mathbb{R}^N$ of the root of $\mathcal{R}$, such that $\|\mathcal{R}(\boldsymbol{x}^*)\| < \text{Tol}$ for a given tolerance and for the Euclidean norm.

1) define an initial guess $\boldsymbol{x}_0$ and set $k = 0$;

2) compute $\mathcal{R}(\boldsymbol{x}_0)$;

3) solve $J(\boldsymbol{x}_k)\delta\boldsymbol{x} = -\mathcal{R}(\boldsymbol{x}_k)$, where $J(\boldsymbol{x}_k)$ is the Jacobian of $\mathcal{R}$ in $\boldsymbol{x}_k$;

4) set $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \delta\boldsymbol{x}$;

5) compute the residual $\mathcal{R}(\boldsymbol{x}_{k+1})$;

6) if $\|\mathcal{R}(\boldsymbol{x}_{k+1})\| < \text{Tol}$, then stop, otherwise increase k and go to 3.

The computational challenge is represented by the inversion of the Jacobian in step 3. When a Krylov subspace method is used, it requires to compute the products of the matrix $J(\boldsymbol{x}_k)$ with many vectors. Moreover the product itself can be costly, even if sometimes it is possible to solve the system inexactly (with a relaxed accuracy). In many cases this cost can be reduced by replacing the Jacobian by a simpler and cheaper operator, as in inexact Newton algorithms. The dynamic preconditioner defined in section 2 can be used to reduce the number of iterations in GMRES iterations. In section 3, we propose a modified Newton algorithm that can be used to further speed up the convergence. In other cases, it is possible or even mandatory to compute the Jacobian in an exact way (see [7]).

3.2 Acceleration strategy

We apply here a strategy proposed by Washio et al. [25] aiming at building a nonlinear subspace acceleration for general nonlinear solvers combined with the replacement of the Jacobian by the preconditioner defined in section 2.3 (noted $\bar{J}$).

The idea in [25] was to store the iterates and the residuals in two different subspaces (of dimension m) which represent a basis for an approximated linear problem. Then they proposed to find a new iterate $\boldsymbol{x}_{\text{new}}$ on the affine subspace $\boldsymbol{x}_k + \sum_{j=0}^{m-1} \alpha_j (\boldsymbol{x}_j - \boldsymbol{x}_k)$ by minimizing the residual. With this goal in mind, the residual is considered affine in this subspace, such that the minimum can be found *without* new evaluation of $\mathcal{R}$.

A simplified acceleration scheme deduced from [25] has been adopted here. We refer to [25] for a complete description of the nonlinear convergence acceleration strategy. This simplified scheme basically requires two parameters: m, the dimension of the minimization subspace and ε_B, a parameter needed to control the nonlinear convergence. We keep in the minimization subspaces the m latest iterates and corresponding residuals when available. In the following, we only detail the procedure to be carried out after a Newton step if convergence is not achieved. Indeed, this can be repeated as long as the approximation $\bar{J}$ of the Jacobian is satisfactory.

Likely, the simplest approximation of the Jacobian is the preconditioner defined in section 2.3, $\bar{J} = P_k \cdots P_1$. This is motivated from the good preconditioning properties in the resolution of the following Jacobian.

1) Find the solution of the minimization problem:

$$\alpha = \arg \min_{\alpha \in \mathbb{R}^m} \left\| \mathcal{R}(\boldsymbol{x}_k) + \sum_{j=0}^{m-1} \alpha_j \left(\mathcal{R}(\boldsymbol{x}_j) - \mathcal{R}(\boldsymbol{x}_k) \right) \right\|;$$

2) Set the candidate iterate as: $\boldsymbol{x}_{\text{new}} = \boldsymbol{x}_k + \sum_{j=0}^{m-1} \alpha_j \left(\boldsymbol{x}_j - \boldsymbol{x}_k \right)$;

3) Control of the acceleration strategy:

$$\begin{aligned}
&\text{if} \quad \varepsilon_B \left\| \boldsymbol{x}_{\text{new}} - \boldsymbol{x}_k \right\| > \min_{j<m} \left\| \boldsymbol{x}_j - \boldsymbol{x}_k \right\| \\
&\qquad \boldsymbol{x}_{\text{new}} = \boldsymbol{x}_k; \\
&\qquad \bar{\mathcal{R}}_{\text{new}} = \mathcal{R}(\boldsymbol{x}_k); \\
&\text{else} \\
&\qquad \bar{\mathcal{R}}_{\text{new}} = \mathcal{R}(\boldsymbol{x}_k) + \sum_{j=0}^{m-1} \alpha_j \left(\mathcal{R}(\boldsymbol{x}_j) - \mathcal{R}(\boldsymbol{x}_k) \right); \\
&\text{end;}
\end{aligned}$$

4) Solve $\bar{J}\delta\boldsymbol{x} = -\bar{\mathcal{R}}_{\text{new}}$ by a Krylov subspace solver;

5) Set the next iterate as: $\boldsymbol{x}_{k+1} = \boldsymbol{x}_{\text{new}} + \delta\boldsymbol{x}$;

6) Compute the next residual: $\mathcal{R}(\boldsymbol{x}_{k+1})$;

7) if $\|\mathcal{R}(\boldsymbol{x}_{k+1})\| < tol$, then stop;

8) Control: if $\|\mathcal{R}(\boldsymbol{x}_{k+1})\| > \min_{j \leqslant m} \|\mathcal{R}(\boldsymbol{x}_j)\|$ then

   (a) Perform a line-search and exit the acceleration step; **or**

   (b) Do not store $\boldsymbol{x}_{k+1}$ and exit the acceleration step;

9) If $\bar{J}$ is not accurate enough, exit the acceleration step, otherwise $k = k + 1$ and go to 1.

Steps 1 to 3 are related to the acceleration scheme and are briefly detailed in the following. To find the minimizing α (item 1), we need to store the points $\boldsymbol{x}_j$ of the Newton algorithm in a matrix V and the values of the residuals $\mathcal{R}(\boldsymbol{x}_j)$ in a matrix F, such that

$$\begin{aligned}
&V = [\boldsymbol{x}_0, \ldots, \boldsymbol{x}_{k-1}] \quad \text{and} \\
&F = [\mathcal{R}(\boldsymbol{x}_0) - \mathcal{R}(\boldsymbol{x}_k), \ldots, \mathcal{R}(\boldsymbol{x}_{k-1}) - \mathcal{R}(\boldsymbol{x}_k)]
\end{aligned} \tag{23}$$

Then α in $\mathbb{R}^k$ can be found by solving the linear problem

$$F^T F\, \alpha = -F^T \mathcal{R}(\boldsymbol{x}_k). \tag{24}$$

Since this problem may be singular, there are two choices: solve the problem with an iterative method (we use GMRES), or, as proposed by Washio et al. in [25], add a term $\epsilon_F Id$ for a small $\epsilon_F = 10^{-16}\max(\text{diag}(F^T F))$ to the matrix $F^T F$ to ensure the invertibility of problem (24) and use a direct method (Cholesky factorization).

Since this scheme is based on a linearization of the residual, this algorithm must be "securised". Thus a control step (step 3) is introduced to avoid the occurrence of too close iterates leading to stagnation in nonlinear convergence. In case of stagnation, both latest

available iterate $\boldsymbol{x}_k$ and residual $\mathcal{R}(\boldsymbol{x}_k)$ are retained. This control criterion can be replaced by a more efficient one, namely $\sum_{j=0}^{m-1} |\alpha_j| > \varepsilon_B$ or $\|\alpha\|_2 > \varepsilon_B$, which in addition can be carried out before step 2.

The positive control parameter ε_B can be chosen depending on the nonlinearity of the problem. If the problem is highly nonlinear, it is crucial to keep ε_B small, since the algorithm is based on the good approximation of the residual in $\boldsymbol{x}_{\text{new}}$ given by the linearization.

Note that this accelerating procedure is rather cheap: a solution of a small system of size $m \times m$, $2m + 2$ inner products and m vector updates and the evaluation of nonlinear residual at most.

Finally a control procedure for the global Newton process (step 8) has been added. Indeed it may happen that the new residual norm $\|\mathcal{R}(\boldsymbol{x}_{k+1})\|$ is larger than the minimal residual norm of the intermediate solutions. There are potentially two reasons for this behavior: either the frozen Jacobian $\bar{J}$ is no more accurate enough or the linearized residual $\bar{\mathcal{R}}_{\text{new}}$ is really different from the true one $\mathcal{R}(\boldsymbol{x}_{new})$. A control procedure is therefore needed to cure these two bottlenecks. In practice however this situation has not been experienced numerically.

In a Newton framework, this acceleration scheme offers two main advantages. First the new residual is expressed as a linear combination of residuals that belong to the minimization subspace of residuals. Thus an expensive operation (evaluation of the residual) is avoided when building the right-hand side for the new Jacobian system. Secondly, the frozen Jacobian $\bar{J}$ can be reused again and again as will be shown in the numerical results. These two points explain the improved efficiency of the preconditioned Newton-GMRES algorithm.

3.3 Global procedure

Finally the global procedure is a preconditioned Newton-GMRES method accelerated by the nonlinear strategy presented in section 3.2. The linear preconditioner is the dynamic preconditioner presented in section 2. This relatively complex algorithm is shown in Figure 1. It is meant to be robust and efficient when treating general (maybe highly) nonlinear problems. As a first evaluation, it has been investigated in the framework of fluid-structure interaction in haemodynamics as described next.

4 Application to fluid-structure interaction in haemodynamics

We present in this section an application of the previous algorithm to fluid-structure interaction problems. We first introduce some notations and we formally recall the general governing equations. We next propose two different approximate tangents of the problem to be used in the inexact Newton algorithms.

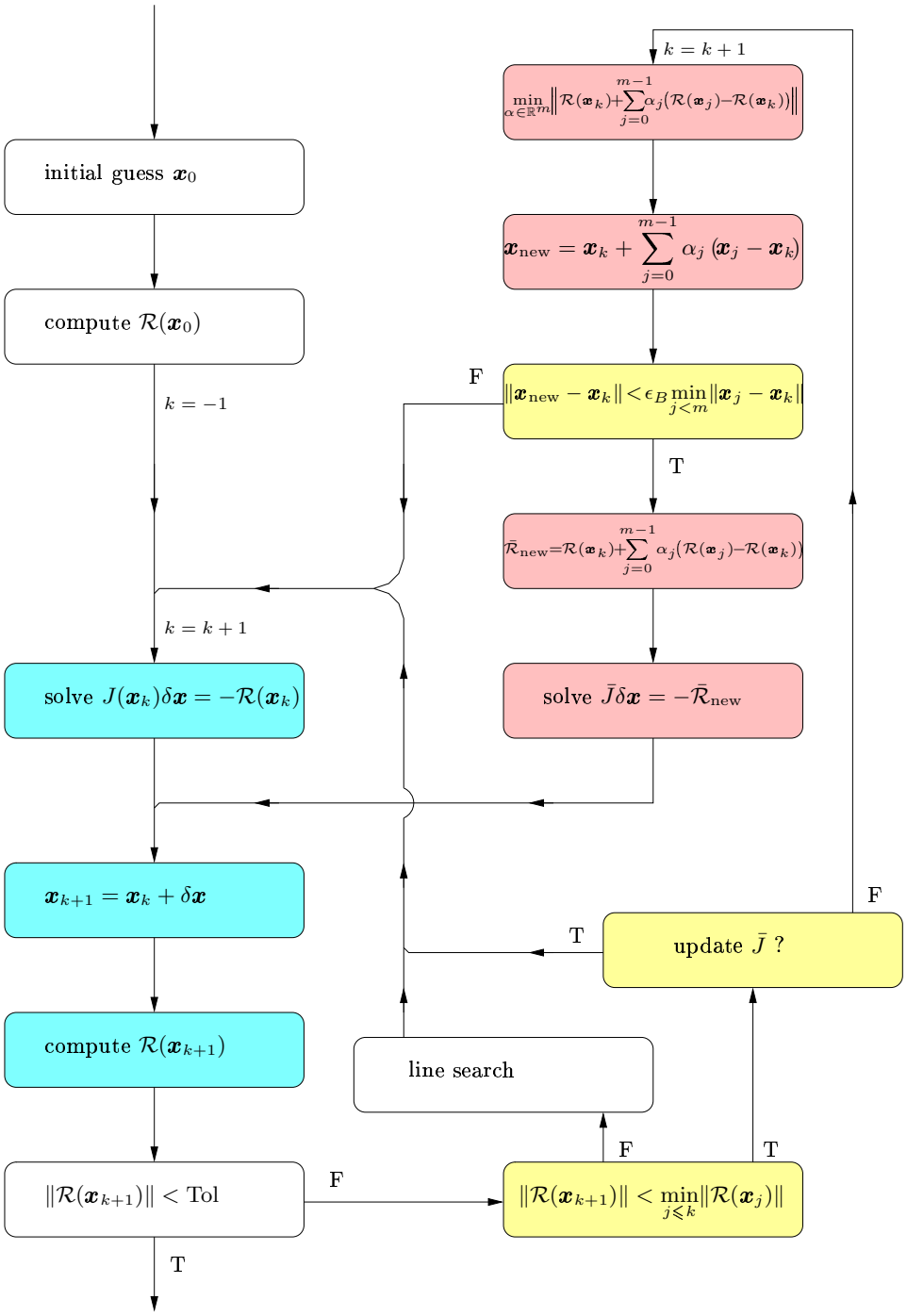

Figure 1: Accelerated Newton algorithm. On the left, the standard Newton algorithm, whereas the nonlinear acceleration strategy is shown on the right.

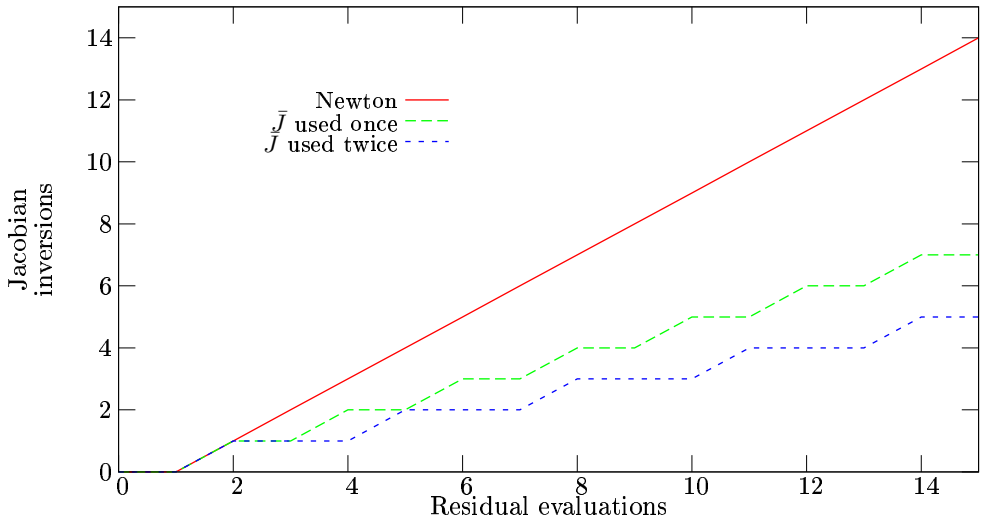

Figure 2: Number of Jacobian inversions per new guess point $\boldsymbol{x}^{k+1}$.

4.1 Problem setting

Since fluid-structure interaction is not the main purpose of this article, we only formally set the equations of the problem, and we refer e.g., to [13], [10], [11] or [6] for a more comprehensive presentation.

We consider the interaction between an incompressible viscous fluid governed by the Navier-Stokes equations and a structure in large displacements. The wet part of the structure, namely the fluid-structure interface, is denoted by $\Sigma(t)$. The fluid domain is moving in order to follow the movements of $\Sigma(t)$. The fluid equations are formulated in the Arbitrary Lagrangian Eulerian (ALE) framework. We suppose that at time t^n, the following quantities are known: The fluid domain Ω_F^n, the fluid velocity $\boldsymbol{u}^n$, the fluid pressure p^n, the displacement of the fluid domain $\hat{\boldsymbol{d}}_F^n$, the displacement $\hat{\boldsymbol{d}}^n$ and the velocity $\hat{\boldsymbol{u}}_S^n$ of the structure. We introduce the following operators (all the known quantities are hidden in these formal notations):

1. For a given displacement of the fluid-structure interface $\hat{\boldsymbol{d}}_\Sigma^{n+1}$, the displacement and the velocity are computed on the whole fluid domain:
$$(\hat{\boldsymbol{d}}_F^{n+1}, \boldsymbol{w}^{n+1}) = \mathcal{D}(\hat{\boldsymbol{d}}_\Sigma^{n+1})$$

2. The Navier-Stokes equations in ALE formulation are solved on the new domain to obtain the velocity and the pressure in the fluid:
$$(\boldsymbol{u}^{n+1}, p^{n+1}) = \mathcal{F}(\hat{\boldsymbol{d}}_F^{n+1}, \boldsymbol{w}^{n+1})$$

3. The force term that the fluid exerts on the structure is computed:

$$\boldsymbol{f}_{\Sigma}^{n+1} = \mathcal{R}(\boldsymbol{u}^{n+1}, p^{n+1})$$

4. The structure equations are solved to obtain the displacement and the velocity in the structure corresponding to the force $\boldsymbol{f}_{\Sigma}^{n+1}$, thus in particular the displacement of the fluid-structure interface:

$$\tilde{\boldsymbol{d}}_{\Sigma}^{n+1} = \mathcal{S}(\boldsymbol{f}_{\Sigma}^{n+1})$$

For the applications we have in mind – blood flow simulations – we are interested in strongly coupled algorithms. This means that a fixed point of the mapping $\mathcal{T} = \mathcal{S} \circ \mathcal{R} \circ \mathcal{F} \circ \mathcal{D}$ has to be found at each time step. Note that $\mathcal{T}$ is a nonlinear operator raised on the fluid-structure interface. The nonlinearities comes from: The acceleration term in the Navier-Stokes equations, the displacements of the fluid domain, the (large) displacements in the structure. For the test cases presented here the constitutive law of the structure is linear.

4.2 Approximation of the Jacobian

It has been illustrated (see e.g., [14], [10]) that fixed-point methods are extremely expensive in situations of practical interest. We now propose two approximate models used to evaluate the Jacobian in an inexact Newton framework. We emphasize that those simplified models are only used to approximate the tangent of the nonlinear problem, the residual being *always* computed with the *complete nonlinear* equations.

FSI-QN 1:

In the first simplified model, we make the following assumptions:

1. The fluid domain is frozen about its current state.
2. The structure equation is linearized about its current state.
3. Nonlinear acceleration and viscous terms are neglected in the fluid. The fluid equation therefore reduces to a Poisson problem on the pressure with Neumann boundary conditions on the interface and Dirichlet ones elsewhere.

FSI-QN 2:

In the second simplified model, we make the following assumptions:

1. The fluid domain is frozen about its current state.
2. The structure equation is linearized about its current state.
3. The fluid equation is linearized about its current state. The fluid problem therefore reduce to an Oseen equation with reaction.

Clearly, FSI-QN1 (proposed in [9, 10]) approximates more roughly the nonlinear problem. It is therefore not surprising that it requires generally more Newton iterations to converge than FSI-QN2. Nevertheless, in all the test cases we have performed, the CPU time was less with FSI-QN1 than with FSI-QN2, each evaluation of FSI-QN1 being much cheaper than FSI-QN2. This conclusion certainly depends on the physical situation considered, and it remains possible that in certain circumstances, FSI-QN2 can lead to better results.

Regardless of the fluid-structure interaction context, the reader may consider that FSI-QN1 corresponds to an approach where the Jacobian of the nonlinear problem is roughly approximated, the convergence therefore requires several, but cheap, Newton iterations, whereas FSI-QN2 corresponds to a better Jacobian approximation, which leads to less numerous, but more expensive, Newton iterations.

4.3 Settings

The linear preconditioner is used in a nested way as explained in section 2.5, and is fully restarted at every time step. A reason for this complete restart is that at a new time step the underlying system can be quite different from the one at the previous time step. As we pointed out, the key idea for the efficiency of this dynamic preconditioner is that the problems in the sequence must be close to each other.

The nonlinear convergence acceleration scheme is used with minimization subspaces of dimension 5 ($m = 5$) and the control parameter ε_B is set to 0.5. The frozen Jacobian $\bar{J}$ corresponds to the previous Jacobian matrix.

4.4 Three-dimensional test case

Test case: The propagation of a pressure wave in a bent cylinder is investigated during 100 time iterations ($\delta t = 2 \cdot 10^{-4}$). The discretized fluid domain has 5874 nodes while the interface and the structure have 1056 nodes.

Algorithm	CPU Time
No Prec.	1=6h16'
Prec.	0.94
Prec. & Acc.	0.88

Table 1: CPU time for the FSI-QN 1 approach.

Algorithm	CPU Time
No Prec.	1=12h36'
Prec.	0.71
Prec. & Acc.	0.73

Table 2: CPU time for the FSI-QN 2 approach.

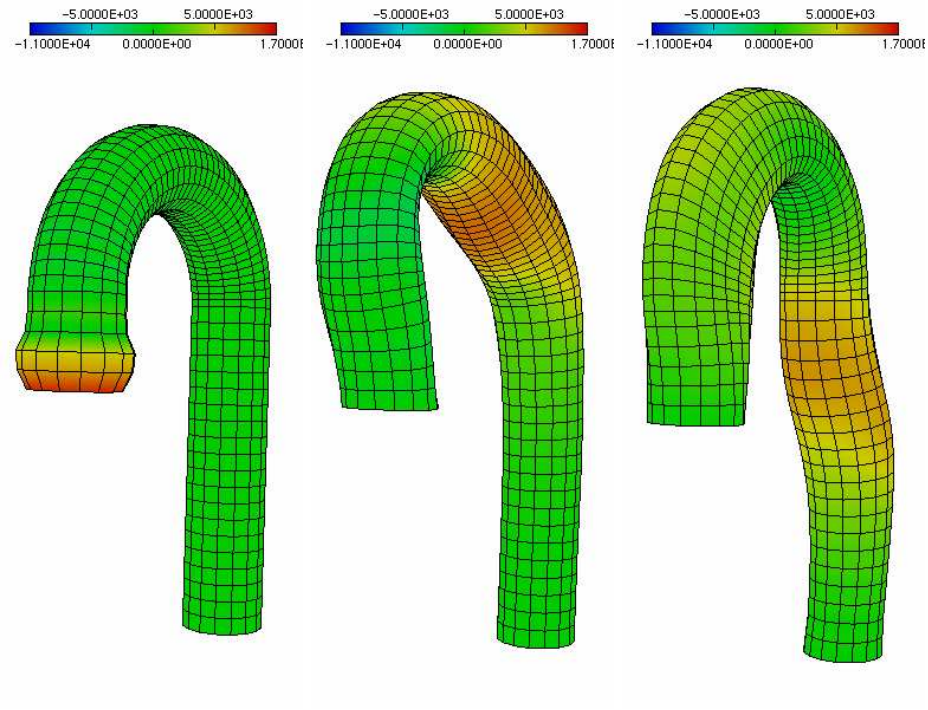

Figure 3: Propagation of a pressure wave in a bent cylinder. Pressure isovalues at different times (see [10] for the details of the test case).

The dynamic preconditioning approach highly reduces the number of GMRES iterations in both experiments (see Figures 4 and 7). However a reduction of only 6 % in CPU time for the first approximation of the tangent problem (see Table 1) is obtained. In the FSI-QN 1 approach, the evaluation of the Jacobian is a small computational task in comparison to the evaluation of the nonlinear residual. Using the dynamic preconditioner reduces the CPU time of 29 % in the second approach (see Table 2). In this experiment the Jacobian evaluation is more involved, hence saving GMRES iterations has a more pronounced impact on the total computing time. In contrast, since the number of Newton iteration is highly reduced (2 or 3 iterations), the accelerated algorithm slows down the computations. In fact, since the approximate Jacobian $\bar{J}$ is not precise enough (with respect to the exact one), the accelerated part of the algorithm is not as efficient as the standard part and the additional cost of residual evaluations slows down the computations.

In Figures 4 and 7 the number of GMRES iterations against the successive Newton resolutions is shown. At each new time step the stored preconditioners are deleted, hence the number of GMRES iterations at the first Newton iteration for the "no prec" and the "prec" approaches are obviously the same. As expected, in both cases the number of Newton iterations is not reduced.

As shown in Figure 2, the acceleration algorithm can be effective when there is at least a relatively high number of Newton iterations to be performed. In fact, in the second experiment, there are only few (2 or 3) Newton iterations at each time step (see Figure 8). Consequently the acceleration approach slows down the computations (see Table 2). On the other hand, in the first experiment the number of Newton iterations to be performed is sufficient for the accelerated algorithm to show its benefits (see Figure 5). In this case the number of genuine Newton iterations is reduced from 8-9 to 3-4. The CPU time is reduced of another 6 % for a total gain of 12 %.

To compare these results with a different problem's size, we have applied the same approach to a two-dimensional case. We report the results in the next section.

4.5 Two-dimensional test case

In this section we show the results of a two-dimensional experiment with the same parameters as the ones of the algorithm considered in the three-dimensional case. The computational domain is a straight tube, the fluid is modeled by the two-dimensional incompressible Navier–Stokes equations and the structure by a generalized string model. The discretized fluid domain has 779 nodes while the interface and the structure have 41 nodes.

The tolerance for the solution of the coupled system is 10^{-6} and 10^{-8} respectively, in order to show the differences when the number of Newton iterations increases.

The results are essentially the same as in three dimensions, except the fact that in this case the approximated Jacobian FSI-QN 2 is only slightly more expensive than FSI-QN 1 and that the improvements of the preconditioner and of the accelerated algorithm in the FSI-QN 1 case are pronounced (see Tables 3 and 4). We also remark that when using a more restrictive tolerance, the approximation with FSI-QN 2 and the accelerated algorithm shows

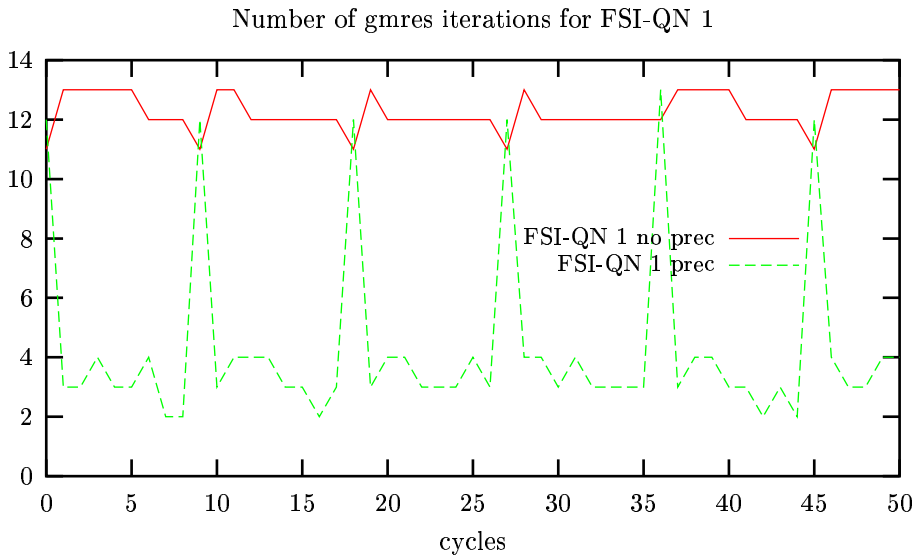

Figure 4: Number of GMRES iterations for FSI-QN 1 (cycles refers to the total number of functional $\mathcal{T}$ evaluations, independenty from the time step). A new time step is characterized by a peak in the number of GMRES iterations. The reason is that the preconditioner is cleared at every new time step.

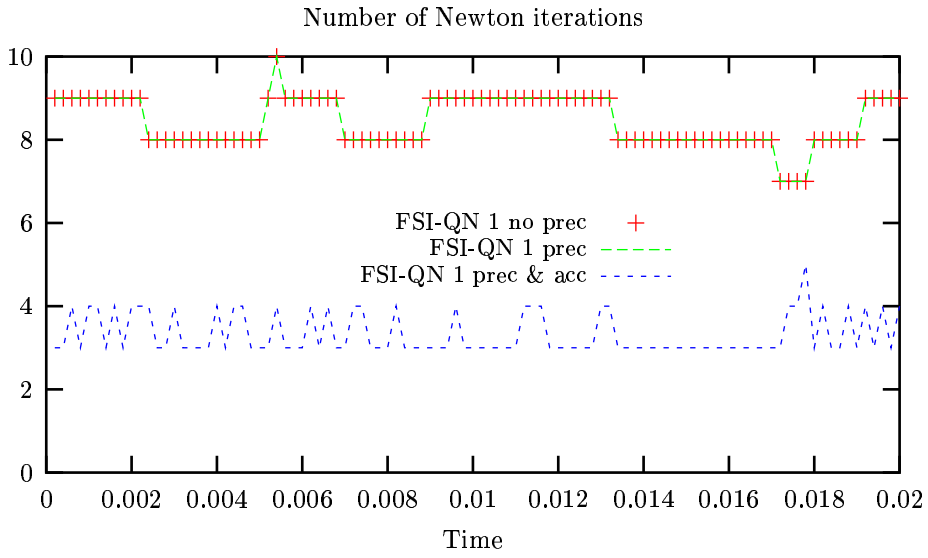

Figure 5: Number of genuine Newton iterations for FSI-QN 1.

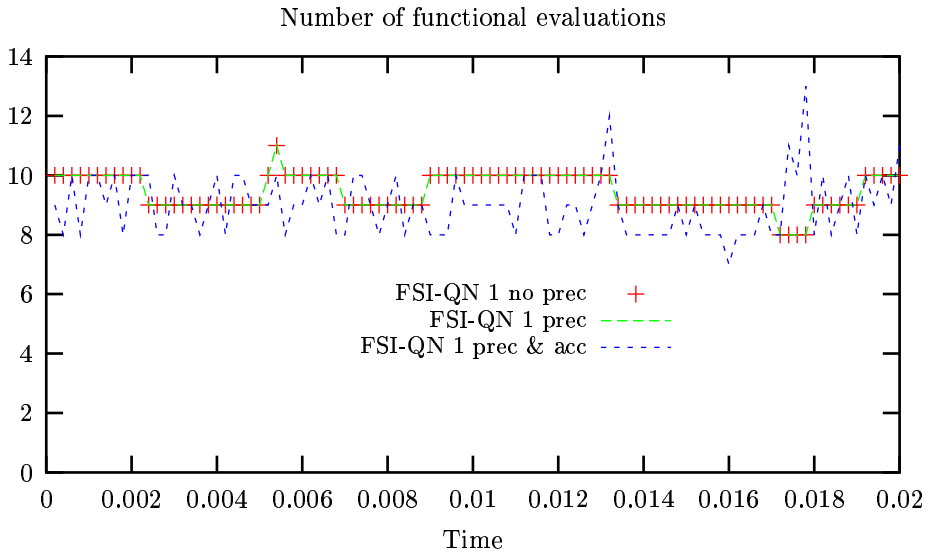

Figure 6: Number of functional $\mathcal{T}$ evaluations for FSI-QN 1.

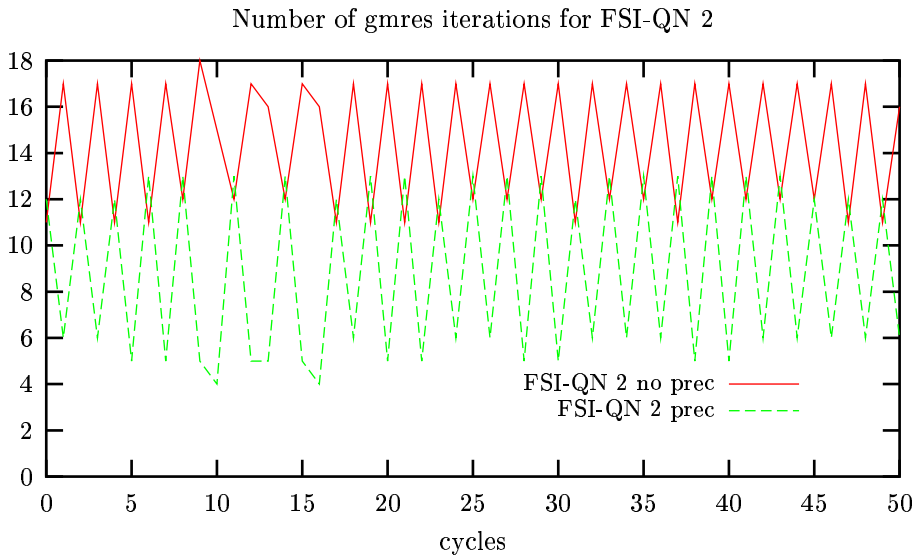

Figure 7: Number of GMRES iterations for FSI-QN 2.

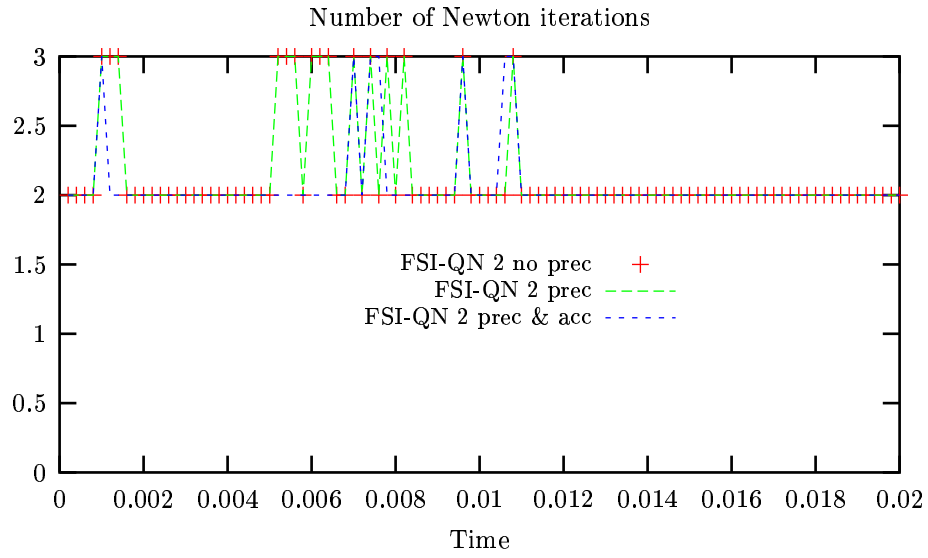

Figure 8: Number of genuine Newton iterations for FSI-QN 2.

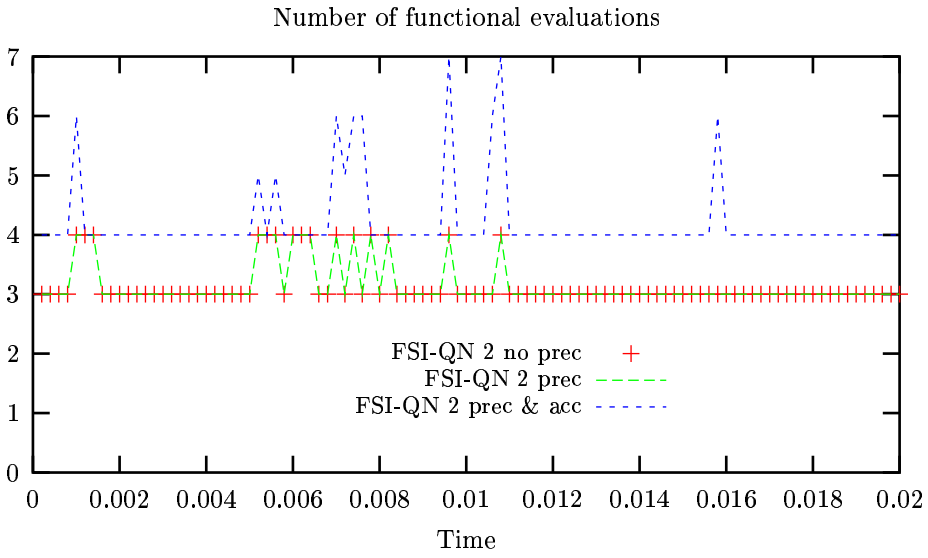

Figure 9: Number of functional $\mathcal{T}$ evaluations for FSI-QN 2.

a performance closer to the FSI-QN 1 approach. The benefits of the presented preconditioner are equivalent in both experiments.

FSI-QN 1	CPU Time	GMRES iter.	residual eval.
No Prec.	1 = 10'09"	1 = 8158	1 = 1219
Prec. P_1	0.78	0.42	1
Nested Prec.	0.77	0.40	1
Prec. & Acc.	0.61	0.21	0.93
FSI-QN 2			
No Prec.	1.13=11'20"	0.39 = 3173	0.49 = 600
Nested Prec.	0.88	0.24	0.49
Prec. & Acc.	0.94	0.24	0.64
FSI-QN 1+2			
Nested Prec	0.87	0.29	0.64

Table 3: Two-dimensional experiment with tolerance 10^{-6} for 200 time steps. The values are normalized with respect to the "no prec." algorithm.

FSI-QN 1	CPU Time	GMRES iter.	residual eval.
No Prec.	1 = 16'35"	1 = 14339	1 = 1942
Prec. P_1	0.72	0.39	0.98
Nested Prec.	0.70	0.34	0.98
Prec. & Acc.	0.50	0.19	0.79
FSI-QN 2			
No Prec.	1.11 = 18'25"	0.39 = 5624	0.41 = 790
Nested Prec.	0.72	0.29	0.41
Prec. & Acc.	0.64	0.15	0.51
FSI-QN 1+2			
Nested Prec.	0.65	0.19	0.46

Table 4: Two-dimensional experiment with tolerance 10^{-8} for 200 time steps. The values are normalized with respect to the "no prec" algorithm.

5 Conclusion

We have proposed a dynamic preconditioner for GMRES to be applied when solving a sequence of linear systems where both the matrix and the right-hand side change. One advantage of this dynamic preconditioner is that it does not require the explicit knowledge of the sequence of matrices A_i. In fact, A_i represents a generic linear operator and in the numerical experiments this matrix has never been built. Only matrix vector products are needed. Therefore this approach is well suited to Jacobian-free Newton-Krylov methods. It has been validated on two- and three-dimensional fluid-structure interaction problems. Gains of up to 29% in CPU time in a three-dimensional experiment and up to 50% in a two-dimensional one have been obtained with this new strategy. The relatively small size of the Jacobian allowed us to use full GMRES as a Krylov subspace method; for larger problems a generalization of this dynamic preconditioner to GMRES(m) is certainly a promising approach.

References

[1] P. Brown and Y. Saad, *Convergence theory of nonlinear Newton-Krylov algorithms*, SIAM Journal on Optimization, 4 (1994), pp. 297–330.

[2] T. Chan and W. Wan, *Analysis of projection methods for solving linear systems with multiple right-hand sides*, SIAM Journal on Scientific Computing, 18- (1997), pp. 1698–1721.

[3] T. F. Chan and M. K. Ng, *Galerkin projection methods for solving multiple linear systems*, SIAM J. Sci. Comput., 21 (1999), pp. 836–850 (electronic).

[4] R. Choquet, *Etude de la méthode de Newton-GMRES. Application aux équations de Navier–Stokes compressible*, PhD thesis, Université de Rennes 1, 1995.

[5] R. Choquet and J. Erhel, *Newton-GMRES algorithm applied to compressible flows*, Internat. J. Numer. Methods Fluids, 23 (1996), pp. 177–190.

[6] S. Deparis, *Numerical Analysis of Axisymmetric Flows and Methods for Fluid-Structure Interaction Arising in Blood Flow Simulation*, PhD thesis, École Polytechnique Fédérale de Lausanne (EPFL), 2004.

[7] M. A. Fernández and M. Moubachir, *A Newton method using exact Jacobians for solving fluid-structure coupling*, INRIA, Rapport de recherche num 5085, (2004).

[8] D. R. Fokkema, G. L. G. Sleijpen, and H. A. van der Vorst, *Accelerated inexact Newton schemes for large systems of nonlinear equations*, SIAM J. Sci. Comput., 19 (1998), pp. 657–674.

[9] J.-F. Gerbeau, *A quasi-Newton method for a fluid-structure problem arising in blood flows*, in Proceedings of the second M.I.T. Conference on Computational Fluid and Solid Mechanics, K. Bathe, ed., Elsevier, 2003, pp. 1355–1357.

[10] J.-F. Gerbeau and M. Vidrascu, *A quasi-Newton algorithm based on a reduced model for fluid structure problems in blood flows*, M2AN, 37 (2003), pp. 631–648.

[11] J.-F. Gerbeau, V. Vidrascu, and P. Frey, *Fluid-structure interaction in blood flows on geometries coming from medical imaging*, tech. report, INRIA No 5052, 2003.

[12] D. Knoll and D. Keyes, *Jacobian-free Newton-Krylov methods: A survey of approaches and applications*, Journal of Computational Physics, 193-2 (2004), pp. 357–397.

[13] P. Le Tallec and J. Mouro, *Fluid structure interaction with large structural displacements*, Comput. Methods Appl. Mech. Engrg., 190 (2001), pp. 3039–3067.

[14] F. Nobile, *Numerical Approximation of Fluid-Structure Interaction Problems with Application to Haemodynamics*, thesis n. 2548, École Polytechnique Fédérale de Lausanne (EPFL), 2001.

[15] M. L. Parks, E. de Sturler, G. Mackey, D. D. Johnson, and S. Maiti, *Recycling Krylov subpaces for sequences of linear systems*, Tech. Report UIUCDCS-R-2004-2421 (CS), Department of Computer Science, University of Illinois at Urbana-Champaign, March 2004 2004.

[16] B. Parlett, *A new look at the Lanczos algorithm for solving symmetric systems of linear equations*, Linear Algebra Appl., 29 (1980), pp. 323–346.

[17] C. Rey and F. Risler, *A Rayleigh-Ritz preconditioner for the iterative solution to large scale nonlinear problems*, Numerical Algorithms, 17 (1998), pp. 279–311.

[18] Y. Saad, *A flexible inner-outer preconditioned GMRES algorithm*, SIAM J. Sci. Comput., 14 (1993), pp. 461–469.

[19] Y. Saad, *On the Lanczos method for solving symmetric linear systems with several right-hand sides*, Math. Comp., 48 (1997), pp. 651–662.

[20] Y. Saad and M. Schultz, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Comput., 7 (1986), pp. 856–869.

[21] V. Simoncini and E. Gallopoulos, *An iterative method for nonsymmetric systems with multiple right-hand sides*, SIAM Journal on Scientific Computing, 16- (1995), pp. 917–933.

[22] C. Smith, A. Peterson, and R. Mittra, *A conjugate gradient algorith for the treatment of multiple incident electromagnetic fields*, IEEE Trans. Antennas and Propogation, 37 (1989), pp. 1490–1493.

[23] H. A. Van der Vorst, *Iterative Krylov methods for large linear systems*, vol. 13 of Cambridge Monographs on Applied and Computational Mathematics, Cambridge University Press, Cambridge, 2003.

[24] B. Vital, *Etude de quelques méthodes de résolution de problèmes linéaires de grande taille sur multiprocesseur.*, PhD thesis, Université de Rennes 1, Rennes, France, 1990.

[25] T. Washio and C. Oosterlee, *Krylov subspace acceleration for nonlinear multigrid schemes*, ETNA, 6 (1997), pp. 271–290.

Unité de recherche INRIA Rocquencourt
Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)

Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)
Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)
Unité de recherche INRIA Sophia Antipolis : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399